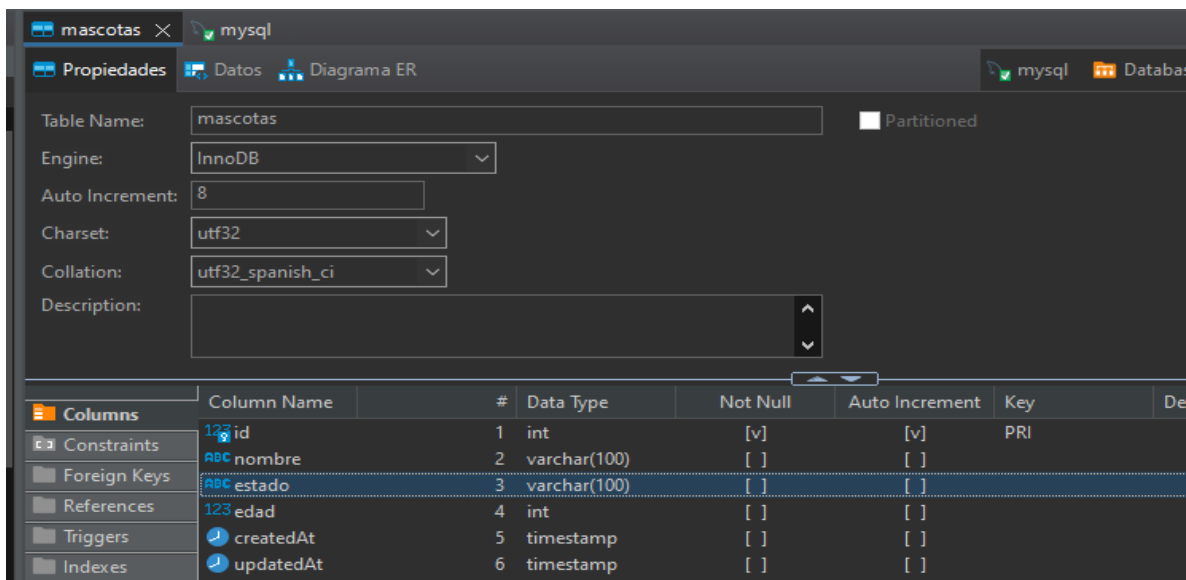
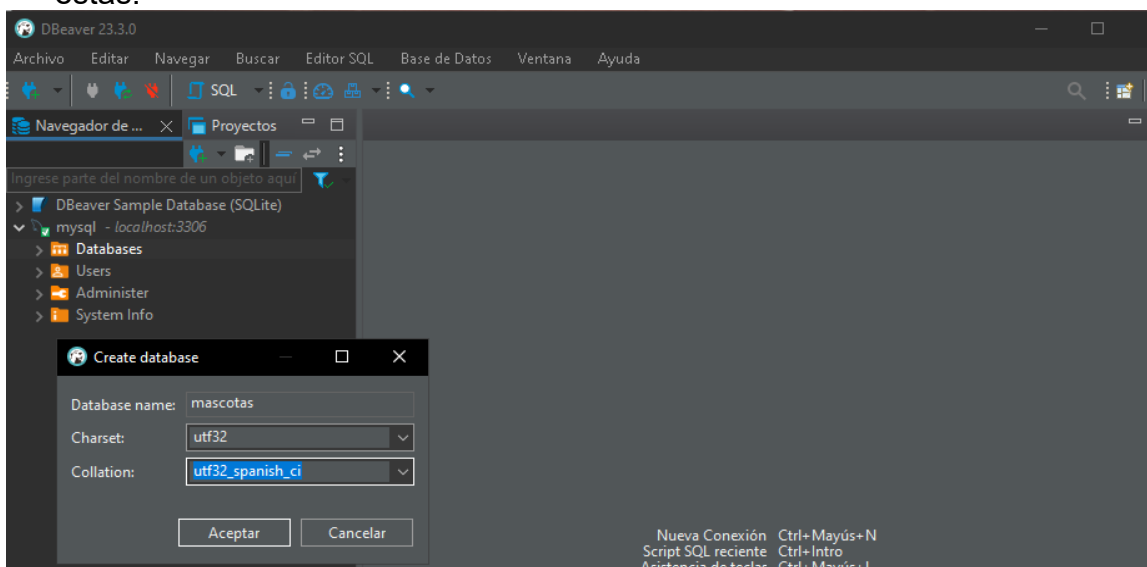


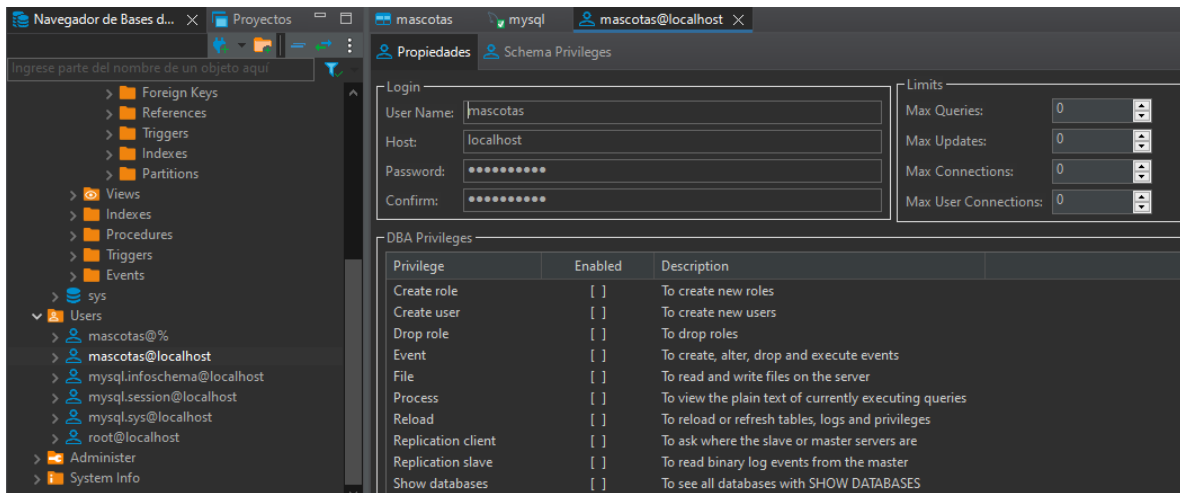
UNIVERSIDAD DE NARIÑO  
FACULTAD DE INGENIERÍA  
PROGRAMA DE INGENIERÍA DE SISTEMAS  
DIPLOMADO DE ACTUALIZACIÓN EN NUEVAS TECNOLOGÍAS PARA EL  
DESARROLLO DE SOFTWARE  
VICENTE AUX REVELO  
HAROLD CALDERON (2140361015)

## TALLER UNIDAD 2 BackEnd

1. Crear una base de datos MYSQL que permita llevar el registro de mascotas (perros y gatos), así como también el proceso de solicitud de adopción de estas.

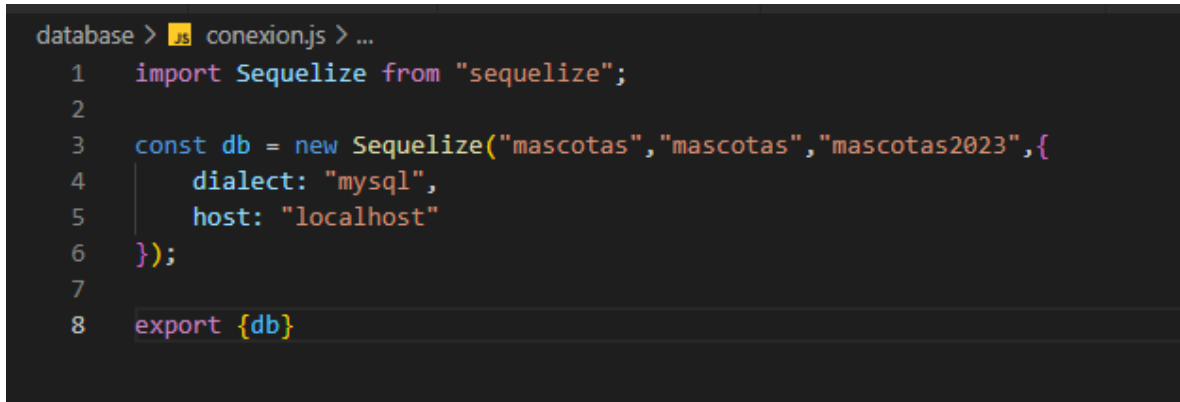


## Creación de Usuario



2. Desarrollar una aplicación Backend implementada en NodeJS y ExpressJS que haga uso de la base de datos del primer punto y que permita el desarrollo de todas las tareas asociadas al registro y administración de las mascotas dadas en adopción por la empresa (La empresa debe contar con un nombre).

## Conexión a Base de Datos



## Desarrollo de tareas

```
app.js package.json mascotasModelo.js mascotasController.js requests.http
controladores > mascotasController.js > crear
1  import {mascotas} from "../modelos/mascotasModelo.js";
2
3  //Crear un recurso
4  const crear = (req,res)=>{
5      if(!req.body.nombre){
6          res.status(400).json({
7              tipo: "error",
8              mensaje: "El nombre no puede estar vacio."
9          });
10     return;
11 }
12 const dataset={
13     nombre: req.body.nombre,
14     edad: req.body.edad,
15     estado: req.body.estado
16 };
17
18 //Usar Sequelize para crear el recurso
19 mascotas.create(dataset).then((resultado)=>{
20     res.status(200).json({
21         tipo: "success",
22         mensaje: "Registro creado correctamente"
23     })
24 })
```

```
controladores > mascotasController.js > crear
35
36 //Buscar recurso por ID
37 const buscarId = (req,res)=>{
38     const id = req.params.id;
39     if(id == null){
40         res.status(203).json({
41             tipo: "error",
42             mensaje: `El id no puede estar vacio`
43         });
44         return;
45     }
46
47     mascotas.findByPk(id).then((resultado)=>{
48         res.status(200).json(resultado);
49     }).catch((err)=>{
50         res.status(500).json({
51             tipo: "error",
52             mensaje: `Registro no encontrado ::: ${err}`
53         });
54     });
55 }
56 }
```


controladores > mascotasController.js > crear

```
69
70 };
71
72 //Actualizar un recurso
73
74 const actualizar=(req,res)=>{
75     const id= req.params.id;
76     if(!req.body.nombre && !req.body.edad){
77         res.status(400).json({
78             mensaje: `No se encontraron Datos para Actualizar`
79         });
80         return;
81     }
82     else{
83         const nombre= req.body.nombre;
84         const edad=req.body.edad;
85         const estado=req.body.estado;
86         mascotas.update({nombre,edad,estado},{where:{id}})
87         .then((resultado)=>{
88             res.status(200).json({
89                 mensaje: `Registro Actualizado`
90             });
91         })
92         .catch((err)=>{
93             res.status(500).json({
94                 mensaje: `Error al actualizar Registro ::: ${err}`
95             });
96         })
97     }
98
99 };
```

```
100
101 //eliminar un recurso
102 const eliminar=(req,res)=>{
103     const id= req.params.id;
104     if(id == null){
105         res.status(203).json({
106             mensaje: `El id no puede estar vacio`
107         });
108         return;
109     }
110     mascotas.destroy({where:{id}})
111     .then((resultado)=>{
112         res.status(200).json({
113             mensaje: `Registro Eliminado`
114         });
115     })
116     .catch((err)=>{
117         res.status(500).json({
118             mensaje: `Error al eliminar Registro ::: ${err}`
119         });
120     })
121
122
123 };
124
125 export {crear,buscarId,buscar,actualizar,eliminar}
```

## Redireccionamiento de pagina

```

rutas >  mascotasRouter.js > ...
1  import express from "express";
2  import {crear, buscarId, buscar, actualizar, eliminar} from "../controladores/mascotasController.js";
3  const routerMascotas = express.Router();
4
5  routerMascotas.get("/", (req, res) => {
6    res.send("Bienvenido a Mascotas");
7  });
8
9  routerMascotas.post("/crear", (req, res) => {
10    crear(req, res);
11  });
12
13  routerMascotas.get("/buscar/:id", (req, res) => {
14    buscarId(req, res);
15  });
16
17  routerMascotas.get("/buscar", (req, res) => {
18    buscar(req, res);
19  });
20
21
22  routerMascotas.put("/actualizar/:id", (req, res) => {
23    actualizar(req, res);
24  });
25
26  routerMascotas.delete("/eliminar/:id", (req, res) => {
27    eliminar(req, res);
28  });
29
30  export {routerMascotas}

```

## Desarrollo de tareas

```
controladores >  mascotasController.js >  crear
1  import {mascotas} from "../modelos/mascotasModelo.js";
2
3  //Crear un recurso
4  const crear = (req,res)=>{
5      if(!req.body.nombre){
6          res.status(400).json({
7              tipo: "error",
8              mensaje: "El nombre no puede estar vacio."
9          });
10         return;
11     }
12     const dataset={
13         nombre: req.body.nombre,
14         edad: req.body.edad,
15         estado: req.body.estado
16     };
17
18     //Usar Sequelize para crear el recurso
19     mascotas.create(dataset).then((resultado)=>{
20         res.status(200).json({
21             tipo: "success",
22             mensaje: "Registro creado correctamente"
23         });
24     }).catch((err)=>{
25         res.status(500).json({
26             tipo: "error",
27             mensaje: `Error al crear el registro ::: ${err}`
28         });
29     })
30 }
```

```

35
36 //Buscar recurso por ID
37 const buscarId = (req,res)=>{
38   const id = req.params.id;
39   if(id == null){
40     res.status(203).json({
41       tipo: "error",
42       mensaje: `El id no puede estar vacio`
43     });
44     return;
45   }
46
47   mascotas.findByPk(id).then((resultado)=>{
48     res.status(200).json(resultado);
49   }).catch((err)=>{
50     res.status(500).json({
51       tipo: "error",
52       mensaje: `Registro no encontrado ::: ${err}`
53     });
54   });
55 }
56
57
58 //Buscar recurso por ID
59 const buscar = (req,res)=>{
60
61   mascotas.findAll().then((resultado)=>{
62     res.status(200).json(resultado);
63   }).catch((err)=>{
64     res.status(500).json({
65       mensaje: `No se encontraron Registros ::: ${err}`
66     });
67   });
68 }
69
70 };

```

```

71
72 //Actualizar un recurso
73
74 const actualizar=(req,res)=>{
75   const id= req.params.id;
76   if(!req.body.nombre && !req.body.edad){
77     res.status(400).json({
78       mensaje: `No se encontraron Datos para Actualizar`
79     });
80     return;
81   }
82   else{
83     const nombre= req.body.nombre;
84     const edad=req.body.edad;
85     const estado=req.body.estado;
86     mascotas.update({nombre,edad,estado},{where:{id}})
87     .then((resultado)=>{
88       res.status(200).json({
89         mensaje: `Registro Actualizado`
90       });
91     })
92     .catch((err)=>{
93       res.status(500).json({
94         mensaje: `Error al actualizar Registro ::: ${err}`
95       });
96     })
97   }
98
99 };

```

```

100
101 //eliminar un recurso
102 const eliminar=(req,res)=>{
103     const id= req.params.id;
104     if(id == null){
105         res.status(203).json({
106             mensaje: `El id no puede estar vacio`
107         });
108         return;
109     }
110     mascotas.destroy({where:{id}})
111     .then((resultado)=>{
112         res.status(200).json({
113             mensaje: `Registro Eliminado`
114         });
115     })
116     .catch((err)=>{
117         res.status(500).json({
118             mensaje: `Error al eliminar Registro ::: ${err}`
119         });
120     })
121
122
123 };
124
125 export {crear,buscarId,buscar,actualizar,eliminar}

```

3. Realizar verificación de las diferentes operaciones a través de un cliente grafico (Postman, Imnsomia, etc.), tomar capturas de pantalla que evidencien el resultado de las solicitudes realizadas.

Verificación a través de Thunder Client

Creación de registro:

The screenshot displays two application windows. The top window is Thunder Client, showing a REST client interface with a POST request to `http://localhost:8000/mascotas/crear`. The response is a JSON object indicating success: `{ "tipo": "success", "mensaje": "Registro creado correctamente" }`. The bottom window is DBeaver 23.3.0, showing a MySQL database named `mascotas`. The `mascotas` table contains the following data:

Id	nombre	estado	edad	createdAt	updatedAt
1	toby	adoptado	2	(NULL)	2023-12-17 23:21:40
2	Lucas	no adoptado	10	2023-12-15 20:37:26	2023-12-17 23:21:49
3	Max	no adoptado	13	2023-12-15 20:57:05	2023-12-17 23:22:16
4	lulu	adoptado	7	2023-12-17 16:31:04	2023-12-17 23:22:25
5	dsad	adoptado	2	2023-12-17 23:19:06	2023-12-17 23:19:06
6	weew	adoptado	8	2023-12-17 23:24:12	2023-12-17 23:24:12
7	13 Princesa	Adoptado	4	2023-12-18 00:37:41	2023-12-18 00:37:41

Actualizar un registro:



PUT http://localhost:8000/mascotas/actualizar/11 Send Status: 200 OK Size: 34 Bytes Time: 41 ms

Query Headers 2 Auth Body 1 Tests Pre Run

JSON XML Text Form Form-encode GraphQL Binary

JSON Content Format

```
1 {
2   "nombre": "Sam",
3   "edad": 2
4 }
```

Response Headers 7 Cookies Results Docs

```
1 {
2   "mensaje": "Registro Actualizado"
3 }
```

DBBeaver 23.3.0 - mascotas

Archivo Editar Navegar Buscar Editor SQL Base de Datos Ventana Ayuda

mysql - localhost:3306

Propiedades Datos Diagrama ER

Enter a SQL expression to filter results (use Ctrl+Space)

	id	nombre	estado	edad	createdAt	updatedAt
1	1	toby	adoptado	2	[NULL]	2023-12-17 23:21:40
2	3	Lucas	no adoptado	10	2023-12-15 20:37:26	2023-12-17 23:21:49
3	4	Max	no adoptado	13	2023-12-15 20:57:05	2023-12-17 23:22:16
4	7	lulu	adoptado	7	2023-12-17 16:31:04	2023-12-17 23:22:25
5	9	dsad	adoptado	2	2023-12-17 23:19:06	2023-12-17 23:19:06
6	11	Sam	adoptado	2	2023-12-17 23:24:12	2023-12-18 00:44:32
7	13	Princesa	Adoptado	4	2023-12-18 00:37:41	2023-12-18 00:37:41
8	14	Princesa	[NULL]	4	2023-12-18 00:42:18	2023-12-18 00:42:18

Eliminar un registro:

DELETE http://localhost:8000/mascotas/eliminar/14 Send Status: 200 OK Size: 32 Bytes Time: 21 ms

Query Headers 2 Auth Body 1 Tests Pre Run

JSON XML Text Form Form-encode GraphQL Binary

JSON Content Format

```
1 {
2   "nombre": "Sam",
3   "edad": 2
4 }
```

Response Headers 7 Cookies Results Docs

```
1 {
2   "mensaje": "Registro Eliminado"
3 }
```

DBBeaver 23.3.0 - mascotas

Archivo Editar Navegar Buscar Editor SQL Base de Datos Ventana Ayuda

mysql - localhost:3306

Propiedades Datos Diagrama ER

Enter a SQL expression to filter results (use Ctrl+Space)

	id	nombre	estado	edad	createdAt	updatedAt
1	1	toby	adoptado	2	[NULL]	2023-12-17 23:21:40
2	3	Lucas	no adoptado	10	2023-12-15 20:37:26	2023-12-17 23:21:49
3	4	Max	no adoptado	13	2023-12-15 20:57:05	2023-12-17 23:22:16
4	7	lulu	adoptado	7	2023-12-17 16:31:04	2023-12-17 23:22:25
5	9	dsad	adoptado	2	2023-12-17 23:19:06	2023-12-17 23:19:06
6	11	Sam	adoptado	2	2023-12-17 23:24:12	2023-12-18 00:44:32
7	13	Princesa	Adoptado	4	2023-12-18 00:37:41	2023-12-18 00:37:41
8	14	Princesa	[NULL]	4	2023-12-18 00:42:18	2023-12-18 00:42:18

Registro id=14 se eliminó

	id	nombre	estado	edad	createdAt	updatedAt
1	1	toby	adoptado	2	[NULL]	2023-12-17 23:21:40
2	3	Lucas	no adoptado	10	2023-12-15 20:37:26	2023-12-17 23:21:49
3	4	Max	no adoptado	13	2023-12-15 20:57:05	2023-12-17 23:22:16
4	7	lulu	adoptado	7	2023-12-17 16:31:04	2023-12-17 23:22:25
5	9	dsad	adoptado	2	2023-12-17 23:19:06	2023-12-17 23:19:06
6	11	Sam	adoptado	2	2023-12-17 23:24:12	2023-12-18 00:44:32
7	13	Princesa	Adoptado	4	2023-12-18 00:37:41	2023-12-18 00:37:41

## Buscar registros:

GET ⌵ http://localhost:8000/mascotas/buscar Send

Query

Headers 2

Auth

Body

Tests

Pre Run

JSON

XML

Text

Form

Form-encode

GraphQL

Binary

JSON Content

Format

1

Status: 200 OK Size: 916 Bytes Time: 23 ms

Response

Headers 7

Cookies

Results

Docs

{}

≡

1

[

2

{

3

"id": 1,

4

"nombre": "toby",

5

"edad": 2,

6

"estado": "adoptado",

7

"createdAt": null,

8

"updatedAt": "2023-12-18T04:21:40.000Z"

9

},

10

{

11

"id": 3,

12

"nombre": "Lucas",

13

"edad": 10,

14

"estado": "no adoptado",

15

"createdAt": "2023-12-16T01:37:26.000Z",

16

"updatedAt": "2023-12-18T04:21:49.000Z"

17

},

18

{

19

"id": 4,

20

"nombre": "Max",

21

"edad": 13,

22

"estado": "no adoptado",

23

"createdAt": "2023-12-16T01:57:05.000Z",

24

"updatedAt": "2023-12-18T04:22:16.000Z"

25

},

26

{

27

"id": 7,

28

"nombre": "lulu",

29

"edad": 7,

30

"estado": "adoptado",

31

"createdAt": "2023-12-17T21:31:04.000Z",

32

"updatedAt": "2023-12-18T04:22:25.000Z"

33

},

34

{

35

"id": 9,

36

"nombre": "dsad",